PERBANDINGAN ALGORITMA K-MEANS DAN K-NEAREST NEIGHBORS PADA SISTEM PERINGKASAN OTOMATIS

Ken Kinanti Purnamasari[1], Nelly Indriani Widiastuti[2]

[1,2] Universitas Komputer Indonesia
Jalan Dipatiukur 112 – 114. Bandung. Indonesia.
E-mail : ken.kinanti@email.unikom.ac.id[1], nelly.indriani@email.unikom.ac.id[2]

ABSTRAK

Peringkasan suatu dokumen mengambil informasi utama yang terkandung dalam dokumen tersebut. Akan tetapi untuk memperoleh informasi penting yang terkandung di suatu artikel, dibutuhkan waktu yang lama. Hal inilah yang menyebabkan munculnya berbagai penelitian yang berkaitan dengan sistem peringkasan otomatis. Dengan adanya sistem ini, pembaca diharapkan dapat lebih mudah menemukan informasi yang relevan dengan kebutuhannya. K-Means dan K-NN adalah dua buah metode yang telah digunakan untuk meringkas teks secara otomatis. Kedua penelitian yang masing-masing menggunakan metode tersebut, menghasilkan kinerja yang baik (akurasi di atas 50%). Namun, untuk dapat digunakan secara luas, perlu diteliti metode mana yang memiliki akurasi lebih tinggi. Berdasarkan hal tersebut dalam penelitian ini, dilakukan perbandingan metode K-Means dan K-NN dalam kasus peringkasan teks secara otomatis. Dokumen yang digunakan sebagai bahan uji adalah dokumen latar belakang laporan Skripsi. Perbandingan dilakukan dengan menggunakan 100 buah data. Berdasarkan pengujian yang telah dilakukan, peringkasan dengan K-NN menghasilkan rata-rata akurasi sebesar 49%, sementara K-Means sebesar 51%. Hal ini menunjukkan bahwa walaupun K-Means memiliki akurasi yang lebih tinggi, perbedaan keduanya tidaklah mencolok secara umum. Dalam beberapa dokumen, K-NN justru menghasilkan akurasi yang lebih tinggi secara signifikan.

Kata kunci : Peringkasan Teks, KNN, K-Means, *Clustering*, *Classification*

1. PENDAHULUAN

1.1 Latar Belakang

Pesatnya perkembangan teknologi online saat ini, menyebabkan adanya kebutuhan informasi yang sangat besar. Sumber informasi seperti artikel-artikel berita yang tersedia secara online, pada umumnya bersifat real time dan tidak terbatas, sehingga membuat sangat banyaknya jumlah kalimat dan paragraf yang terkandung di dalamnya.[1] Maka, untuk melakukan proses pembacaan secara menyeluruh dalam memperoleh informasi inti dari suatu artikel berita, diperlukan waktu yang sangat lama. Hal inilah yang memicu dikembangkannya penelitian tentang sistem peringkasan otomatis. Dengan adanya sistem ini, manusia diharapkan dapat lebih mudah menemukan informasi yang relevan dengan kebutuhannya, karena informasi yang penting telah diseleksi oleh sistem ini.

Beberapa penelitian tentang peringkasan teks otomatis dalam berbagai bahasa telah dilakukan. Salah satunya dilakukan oleh Fendra Pratama pada tahun 2014. Fendra menggunakan K-means clustering untuk meringkas dokumen tunggal berita artikel dalam bahasa Indonesia. Pada kompresi data sebesar 40%, penelitian tersebut menghasilkan recall 60%, precision 62%, dan f-measure 61%.[2] Sementara penelitian lainnya dalam kasus peringkasan bahasa Inggris yang menggunakan K-Nearest Neighbors (K-NN) dapat menghasilkan recall 63%, precision 100%, dan f-measure 77%. [3]

Kedua penelitian tersebut menyatakan bahwa K-Means dan KNN memiliki kinerja yang cukup baik dalam melakukan peringkasan teks. Namun, untuk dapat digunakan dalam melakukan peringkasan teks, perlu diketahui metode mana yang memiliki performa lebih tinggi. Maka, penelitian ini membandingkan kedua metode tersebut dalam kasus peringkasan teks laporan Skripsi.

1.2 Maksud dan Tujuan

Maksud dari penelitian ini adalah untuk membangun sebuah sistem peringkasan otomatis yang menggunakan algoritma K-Means dan K-NN. Sementara yang menjadi tujuan dari penelitian ini adalah mengetahui akurasi metode terbaik diantara K-means dan K-NN dalam kasus peringkasan teks laporan Skripsi.

2. ISI PENELITIAN

2.1 Landasan Teori

2.1.1 Peringkas Teks Otomatis

Peringkas Teks Otomatis adalah suatu sistem yang menghasilkan suatu teks yang berisi informasi yang penting dari suatu teks sumber.[4] Sistem tersebut dibangun untuk memperoleh informasi inti

dari suatu teks tanpa menghilangkan kandungan topik utama, sehingga dapat membantu memperkecil jumlah waktu yang dibutuhkan oleh pengguna untuk membaca suatu teks.[5]

Berdasarkan keluaran yang dihasilkannya, peringkas teks otomatis dapat dibagi menjadi dua tipe, yaitu ekstraktif dan abstraktif. Ekstraktif menghasilkan ringkasan dari sebagian kalimat yang terdapat dalam teks sumber (biasanya dilakukan dengan metode statistik, linguistik, dan heuristik, atau kombinasi ketiganya), sementara Abstraktif menginterpretasi teks sumber dan dapat menghasilkan ringkasan yang tidak persis sama dengan teks sumber. Penelitian ini menggunakan konsep ekstraktif, sehingga akurasi dapat dihasilkan dengan membandingkan kalimat-kalimat yang menjadi ringkasan sistem dan ringkasan manual yang divalidasi oleh ahli bahasa. [6][7]

2.1.2 Praproses

Penelitian ini menggunakan tiga buah praproses teks yang meliputi Tokenizing, Case Folding, dan Filtering. Tokenizing umumnya dipahami sebagai proses membentuk kata-kata dari urutan karakter dalam suatu dokumen. Biasanya proses ini dilakukan dengan mendeteksi kumpulan karakter alphanumeric berukuran lebih dari tiga buah, yang diakhiri oleh spasi atau suatu karakter spesial. [8] Namun, dalam penelitian ini, proses pembuatan token dilakukan terhadap pembentukan kalimat terlebih dahulu. Hal ini dilakukan untuk menunjang tahap selanjutnya yang menghitung kemunculan kata di setiap kalimat. Maka, dalam penelitian ini terdapat dua tahap tokenizing, yaitu Tokenizing kata dan Tokenizing kalimat. Case Folding merupakan proses penyamaan karakter huruf menjadi huruf besar atau huruf kecil. [9] Tahap ini dilakukan untuk memperkecil lingkup pemeriksaan teks, sehingga huruf besar dan huruf kecil dianggap sama. Dalam penelitian ini, case folding mengubah seluruh karakter huruf menjadi huruf kecil. Filtering menghapus berbagai karakter yang tidak digunakan. Dalam penelitian ini, karakter-karakter yang dihapus merupakan karakter di luar huruf dan titik(.). [9]

2.1.3 Pembobotan TF-IDF

TF-IDF merupakan salah satu cara yang dapat digunakan untuk memberikan bobot terhadap setiap kata yang ada di dalam teks. [10] Bobot ini diperoleh dengan menghitung frekuensi kemunculan suatu term (kata) di suatu dokumen (Term Frequency/TF) dan kemungkinan suatu dokumen memiliki suatu kata (Inverse Document Frequency/IDF). Beberapa variasi dari skema pembobotan TF-IDF sering digunakan dalam kasus information retrieval. Kemungkinan suatu dokumen memiliki suatu kata dapat dihitung dengan rumus yang terdapat pada Persamaan 1 berikut.

$$IDF = \log\left(\frac{N}{df}\right) \qquad (1)$$

Dimana N adalah jumlah dokumen yang berisi term (t) dan df adalah jumlah kemunculan (frekuensi) term terhadap D. Adapun algoritma yang digunakan untuk menghitung bobot (w) masing-masing dokumen terhadap kata kunci (query) menggunakan Persamaan 2 di bawah ini.

$$w_{d.t} = tf_{d.t} * IDF_t \qquad (2)$$

Dimana D adalah indeks dokumen, t adalah indeks dari kata, tf adalah frekuensi kemunculan kata dalam suatu kalimat, dan w_(d.t) adalah bobot dokumen ke-d terhadap term ke-t.

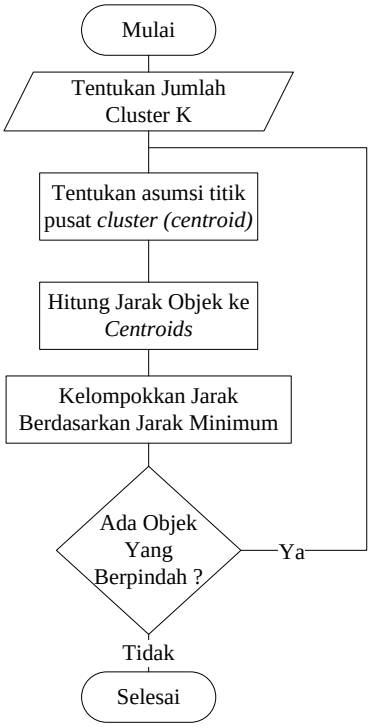
2.1.4 Cosine Similarity

Cosine Similarity memodelkan teks ke dalam suatu vektor term dan menghitung nilai cosine dari dua buah term dalam vektor yang terbentuk. Term yang digunakan dapat berupa kata, paragraf, atau bahkan keseluruhan teks sumber. Setelah melalui perhitungan, biasanya bobot Cosine akan diurutkan. Nilai cosine yang tinggi menunjukkan tingkat kesamaan yang besar. Dalam penelitian ini, Cosine Similarity dihitung dengan menggunakan Persamaan 3, dimana t menunjukkan indeks penunjuk term kalimat, Wt.b1 menunjukkan bobot term t dalam blok b1, dan Wt.b2 menunjukkan bobot term t dalam blok b2. [11]

$$CS(b_1, b_2) = \frac{\sum_{t=1}^{n} W_{t.b1} W_{t.b2}}{\sqrt{\sum_{t=1}^{n} W_{t.b1}^2} \sqrt{\sum_{t=1}^{n} W_{t.b2}^2}} \qquad (3)$$

2.1.5 K-Means

K-Means merupakan salah satu metode unsupervised learning. [12] Metode K-Means termasuk ke dalam kategori clustering non hirarki yang mudah dan sederhana, cepat menemukan konvergensi, dan dapat beradaptasi terhadap sebaran data. [13] Metode ini membentuk cluster dengan cara menghitung nilai kedekatan dari antara suatu data dengan pusat dari setiap cluster. Alur proses dari K-Means yang digunakan dalam penelitian ini dapat dilihat pada Gambar 1 berikut.



Gambar 1. Alur Proses Algoritma K-Means

Penentuan jumlah cluster pada penelitian ini disesuaikan dengan kasus peringkasan yang akan dilakukan. Pengelompokan dapat dilakukan untuk membuang kalimat yang tidak cukup penting, didasarkan pada relevansi-nya dengan keseluruhan isi berita. Hal ini memungkinkan digunakannya dua buah cluster.

Penentuan nilai centroid dilakukan secara acak di awal iterasi. Sedangkan nilai centroid selanjutnya akan diperoleh dengan menggunakan nilai rata-rata dari kumpulan data yang memiliki centroid yang sama.

Penghitungan jarak antara data dan pusat cluster dilakukan dengan Euclidean Distance.[12] Euclidean digunakan dengan menghitung jarak antara suatu data dengan pusat cluster awal. Dengan perhitungan ini, jarak terpendek yang menentukan kedekatan data dengan suatu cluster dapat ditentukan. Euclidean Distance yang digunakan dalam penelitian ini, dihitung menggunakan Persamaan 4.

$$Ed(x_i, y_j) = \sqrt{(x_i - y_j)^2} \qquad (4)$$

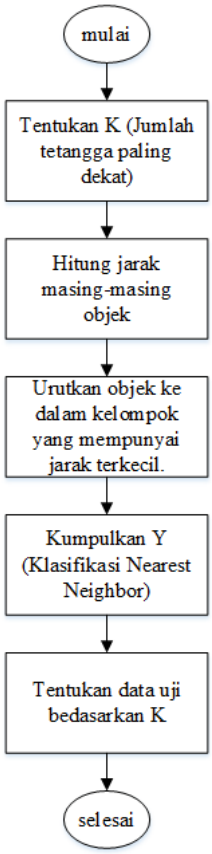
Dimana $Ed(x_i, y_i)$ adalah euclidean distance, x_i sebagai bobot dokumen ke-i, dan y_j sebagai pusat cluster ke-j.

Penentuan anggota suatu cluster dilakukan dengan memperhitungkan jarak terpendek yang telah dihitung sebelumnya. Suatu data akan dimasukkan ke dalam suatu cluster, jika jarak antara data dan cluster tersebut merupakan jarak terpendek. Dari pengelompokkan ini, dapat terlihat pola sebaran data ke dalam setiap cluster yang terbentuk.

Proses akan diulang kembali ke tahap Penentuan nilai centroid, selama pola centroid yang dihasilkan masih berubah-ubah. Dalam kasus tertentu, untuk mengatasi data yang tidak menghasilkan pola yang sama, penelitian ini menggunakan batas iterasi maksimum, sebanyak 100 kali.

2.1.6 K-NN

KNN adalah salah satu metode instance-based yang paling sederhana. Dalam penelitian ini, digunakan Traditional KNN yang menggunakan bobot yang telah diperoleh dari hasil perhitungan Cosine Similarity atau TF-IDF untuk menentukan k buah tetangga terdekat.[14][15] Sebagai algoritma clustering, KNN melakukan klasifikasi berdasarkan kelas yang telah ditentukan sebelumnya. Dalam peringkasan di penelitian ini, kelas dari tetangga ditentukan berdasarkan ambang batas tertentu (rata-rata bobot similaritas). Alur proses dari KNN digambarkan dalam Gambar 2 berikut.

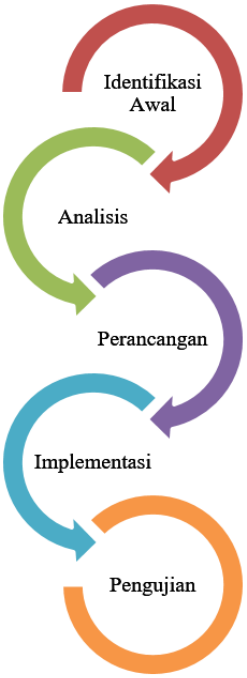


Gambar 2. Alur Proses Algoritma K-NN

2.2 Metode Penelitian

Metode penelitian yang digunakan terdiri dari empat langkah utama, yaitu Identifikasi Awal, Analisis, Perancangan, Implementasi, dan Pengujian (Gambar 3). Identifikasi Awal mencakup pengumpulan data, informasi, dan literatur yang terkait dengan Sistem Peringkasan Otomatis, Algoritma K-Means, dan Algoritma K-NN. Analisis mencakup proses pengamatan pola data, pemilihan

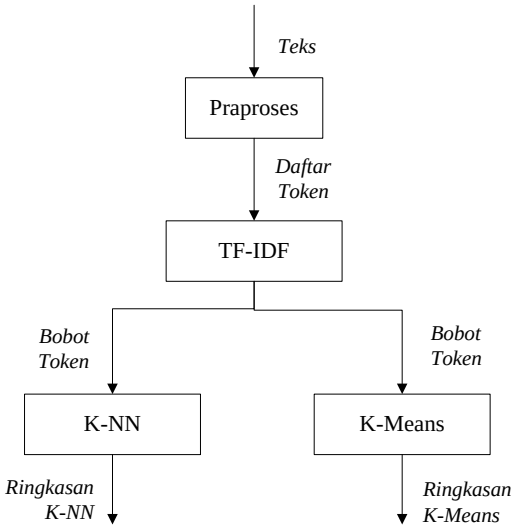
metode, dan penyesuaian parameter. Sementara Perancangan mencakup proses mensimulasikan sistem peringkasan otomatis sehingga memiliki gambaran yang jelas di setiap tahapnya. Pada tahap Implementasi, mulai dilakukan pengkodean aplikasi Sistem Peringkas Otomatis yang menerapkan Algoritma K-Means dan K-NN. Tahap terakhir dari penelitian ini adalah pengujian, dimana dokumen ringkasan hasil keluaran dari sistem yang menggunakan dua algoritma tersebut akan dibandingkan akurasinya.



Gambar 3. Alur Proses Penelitian

2.3 Hasil dan Pembahasan

Peringkasan teks dengan menggunakan KNN dan K-Means menggunakan tahapan praproses yang sama. Berikut adalah arsitektur sistem dari sistem peringkasan otomatis yang akan dibangun. Sesuai dengan alur sistem yang terdapat pada Gambar 4, terdapat 4 proses yang dilakukan dalam penelitian ini, yaitu Praproses, TF-IDF, K-NN, dan K-Means.



Gambar 4. Arsitektur Sistem

2.3.1 Data Masukan

Dokumen masukan diambil dari Latar Belakang pada laporan Skripsi Teknik Informatika UNIKOM (http://elib.unikom.ac.id). Contoh data yang digunakan dapat dilihat pada Gambar 5.

> Pengenalan suara merupakan proses identifikasi suara berdasarkan kata yang diucapkan seseorang yang ditangkap oleh perangkat input suara untuk dikenali dan kemudian diterjemahkan menjadi sebuah data yang dipahami oleh komputer. Ketika manusia mengeluarkan suara, saat itulah suara tersebut menyampaikan beberapa informasi di dalam kata yang diucapkan melalui gelombang suara. Informasi suara tersebut dapat diketahui melalui fitur suara itu sendiri, diantaranya dengan pitch dan formant, dimana pitch adalah frekuensi fundamental dari sinyal suara yang dihasilkan karena getaran pita suara, dan formant adalah frekuensi resonansi akustik dari bidang suara manusia [1]. Kedua fitur tersebut merupakan fitur suara yang sangat penting untuk mengidentifikasi suara yang diucapkan dari seseorang [2][3]. Pada penelitian sebelumnya, Yixion Pan, Peipei Shen, dan Liping Shen melakukan pengenalan emosi suara menggunakan Support Vector Machine. Mereka menganalisis emosi suara dari bahasa Jerman dan China, serta mengkombinasikan ekstraksi fitur dengan MFCC+MEDC+Energy menghasilkan akurasi emosi suara bahasa China sebesar 91,3% dan emosi suara bahasa Jerman sebesar 95,1% [5]. Metode Support Vector Machine (SVM) merupakan metode klasifikasi jenis terpandu karena ketika proses pelatihan, diperlukan target pembelajaran tertentu. Meskipun waktu pelatihan SVM kebanyakan lambat, tetapi metode ini sangat akurat karena kemampuannya untuk menangani model-model nonlinear yang kompleks. SVM dapat digunakan untuk prediksi dan klasifikasi [4]. Berdasarkan penelitian yang telah dilakukan sebelumnya, maka dalam skripsi ini akan membuat simulator pengidentifikasian suara tinggi dan rendah pada pria dan wanita ditinjau dari pitch dan formant menggunakan metode Support Vector Machine.

Gambar 5. Contoh Teks Masukan

2.3.2 Praproses Teks

Tahap praproses terdiri dari empat buah subproses, yaitu Filtering, Tokenizing Kalimat, Case Folding, Tokenizing Kata, dan TF-IDF (Gambar 6).

Gambar 6. Alur Praproses

Filtering menghapus karakter-karakter di luar huruf dan titik (.). Sebagai contoh, bagian pemanggilan referensi yang mengandung angka dan kurung siku (Gambar 7) akan digantikan dengan spasi (Gambar 8).

> …
> Kedua fitur tersebut merupakan fitur suara yang sangat penting untuk mengidentifikasi suara yang diucapkan dari seseorang **[2][3]**.
> …

Gambar 7. Teks Awal

> …
> Kedua fitur tersebut merupakan fitur suara yang sangat penting untuk mengidentifikasi suara yang diucapkan dari seseorang .
> …

Gambar 8. Teks Tanpa Simbol

Tokenizing kalimat akan memecah teks tanpa simbol menjadi token-token kalimat. Pemecahan teks ini dilakukan dengan menggunakan titik (.) sebagai penanda akhir dari suatu kalimat. Contoh daftar token kalimat yang dihasilkan oleh proses ini dapat dilihat pada Tabel 1 di bawah ini.

Tabel 1. Daftar Token Kalimat

S_i	Kalimat
S_1	Pengenalan suara merupakan proses identifikasi suara berdasarkan kata yang diucapkan seseorang yang ditangkap oleh perangkat input suara untuk dikenali dan kemudian diterjemahkan menjadi sebuah data yang dipahami oleh komputer
S_2	Ketika manusia mengeluarkan suara saat itulah suara tersebut menyampaikan beberapa informasi di dalam kata yang diucapkan melalui gelombang suara
S_3	Informasi suara tersebut dapat diketahui melalui fitur suara itu sendiri diantaranya dengan pitch dan formant dimana pitch adalah frekuensi fundamental dari sinyal suara yang dihasilkan karena getaran pita suara dan formant adalah frekuensi resonansi akustik dari bidang suara manusia
S_4	Kedua fitur tersebut merupakan fitur suara yang sangat penting untuk mengidentifikasi suara yang diucapkan dari seseorang
S_5	Pada penelitian sebelumnya Yixion Pan Peipei Shen dan Liping Shen melakukan pengenalan emosi suara menggunakan Support Vector Machine
S_6	Mereka menganalisis emosi suara dari bahasa Jerman dan China serta mengkombinasikan ekstraksi fitur dengan MFCC MEDC Energy menghasilkan akurasi emosi suara bahasa China sebesar dan emosi suara bahasa Jerman sebesar
S_7	Metode Support Vector Machine SVM merupakan metode klasifikasi jenis terpandu karena ketika proses pelatihan diperlukan target pembelajaran tertentu
S_8	Meskipun waktu pelatihan SVM kebanyakan lambat tetapi metode ini sangat akurat karena kemampuannya untuk menangani modelmodel nonlinear yang kompleks
S_9	SVM dapat digunakan untuk prediksi dan klasifikasi

Case Folding dalam penelitian ini melakukan penyeragaman teks menjadi huruf kecil. Sebagai contoh, pada token kalimat pertama, karakter awal yang merupakan huruf kapital (Gambar 9) akan diubah menjadi huruf kecil (Gambar 10).

Pengenalan suara merupakan proses identifikasi suara berdasarkan kata yang diucapkan seseorang yang ditangkap oleh perangkat input suara untuk dikenali dan kemudian diterjemahkan menjadi sebuah data yang dipahami oleh komputer

Gambar 9. Token Kalimat

pengenalan suara merupakan proses identifikasi suara berdasarkan kata yang diucapkan seseorang yang ditangkap oleh perangkat input suara untuk dikenali dan kemudian diterjemahkan menjadi sebuah data yang dipahami oleh komputer

Gambar 10. Token Kalimat dengan Case Folding

Tokenizing kata memecah setiap token kalimat menjadi token kata. Pemecahan ini dilakukan dengan menggunakan spasi sebagai penanda akhir dari suatu kata. Contoh daftar token kata dari token kalimat S_1 dapat dilihat pada Tabel 2 di bawah ini.

Tabel 2. Daftar Token Kata

W_i	Kata	W_i	Kata
W_1	pengenalan	W_{16}	input
W_2	suara	W_{17}	suara
W_3	merupakan	W_{18}	untuk
W_4	proses	W_{19}	dikenali
W_5	identifikasi	W_{20}	dan
W_6	suara	W_{21}	kemudian
W_7	berdasarkan	W_{22}	diterjemahkan
W_8	kata	W_{23}	menjadi
W_9	yang	W_{24}	sebuah
W_{10}	diucapkan	W_{25}	data
W_{11}	seseorang	W_{26}	yang
W_{12}	yang	W_{27}	dipahami
W_{13}	ditangkap	W_{28}	oleh
W_{14}	oleh	W_{29}	komputer
W_{15}	perangkat		

2.3.3 TF-IDF

Pada tahap ini, bobot setiap kata dihitung berdasarkan frekuensi kemunculannya. Sebelum bobot TF-IDF dapat diperoleh, perlu diketahui terlebih dulu nilai dari TF, DF, dan IDF. Sebagai contoh, kata "pengenalan" yang hanya muncul pada kalimat 1 dan 5 saja, menghasilkan TF dengan nilai 1 untuk S_1 dan S_5, serta nilai 0 untuk kalimat lainnya. DF diisi dengan kemunculan kata "pengenalan" pada setiap dokumen, dengan menghitung jumlah total dari kemunculan kata dalam seluruh dokumen. Kata "pengenalan" yang hanya muncul pada dua kalimat (1 dan 5) akan menghasilkan DF dengan nilai 2. Dengan jumlah kalimat sebanyak 10 buah, maka dapat ditentukan bahwa nilai N = 10. Maka, IDF dapat dihitung dengan menggunakan perhitungan pada Gambar 11.

$$\begin{aligned} IDF &= \log\left(\frac{N}{df}\right) \\ &= \log\left(\frac{10}{2}\right) \\ &= \log 5 \\ &= 0.699 \end{aligned}$$

Gambar 11. Perhitungan IDF

Setelah nilai TF dan IDF tersedia, maka nilai w diperoleh dengan mengalikan TF dan IDF dari token "pengenalan". Contoh perhitungannya dapat dilihat pada Gambar 12 di bawah ini.

$$\begin{aligned} w &= TF * IDF \\ &= 1 * 0.699 \\ &= 0.699 \end{aligned}$$

Gambar 12. Perhitungan TF-IDF

Maka, hasil perhitungan TF-IDF untuk token kata "pengenalan" di setiap kalimat, dapat dilihat pada Tabel 3 di bawah ini. Dapat terlihat bahwa w pada suatu kalimat akan bernilai lebih dari 0 ketika token terdapat pada kalimat tersebut.

Tabel 3. TF-IDF "Pengenalan"

S_i	*w "pengenalan"*
S_1	0.699
S_2	0
S_3	0
S_4	0
S_5	0.699
S_6	0
S_7	0
S_8	0
S_9	0
S_{10}	0

Setelah bobot untuk setiap token kata dihitung, maka total w dapat digunakan untuk menghasilkan nilai vektor seperti yang dapat dilihat pada Tabel 4 untuk setiap kalimat. Nilai vektor ini merupakan hasil akar kuadrat dari kuadrat bobot TF.IDF.

Tabel 4. Hasil Perhitungan Vektor

S_i	Total w^2	Vektor
S_1	17.7075	4.2080
S_2	11.0667	3.3267
S_3	24.2373	4.9231
S_4	5.9488	2.4390
S_5	10.4408	3.2312
S_6	21.0725	4.5905
S_7	10.2281	3.1981
S_8	13.3313	3.6512
S_9	3.4089	1.8463
S_{10}	18.1032	4.2548

2.3.4 K-Means

K-Means pada penelitian ini menggunakan bobot vektor dari TF-IDF. Penentuan jumlah cluster dilakukan dengan rumus dalam Persamaan 5. Dikarenakan teks input yang digunakan terdiri dari 10 buah kalimat, maka akan diperoleh nilai k = 2. Jika hasil pembagian memiliki nilai di belakang koma, maka dilakukan pembulatan ke bawah.

$$k = \sqrt{n/2} \qquad (5)$$

Kemudian, untuk memulai pengelompokkan, ditentukan nilai awal centroid. Pemilihan ini dilakukan secara acak. Misalnya terpilih nilai centroid dari S_9 dan S_4, maka nilai awal akan diambil dari nilai vektor kalimat ke-9 dan ke-4 sebagai berikut.

Pusat *cluster* ke-1 (wC_1) = 1.8463
Pusat *cluster* ke-2 (wC_2) = 2.4390

Setiap nilai bobot akan dihitung jaraknya dengan pusat cluster untuk menentukan kelompoknya. Jarak ditentukan dengan menggunakan rumus Euclidean Distance pada Persamaan 6.

$$ed(wC_i, w_j) = \sqrt{(wC_i - w_j)^2} \qquad (6)$$

Hasil akhir yang terbentuk setelah diperoleh pola sebaran cluster yang sama, pada iterasi ke-3, dapat dilihat pada Tabel 5 di bawah ini.

Tabel 5. Cluster yang terbentuk

Cluster	Total
C_1	0

C_2	23.8383

2.3.5 K-NN

Dalam penelitian ini, K-NN menentukan tetangga terdekat dari setiap kalimat berdasarkan nilai kesamaan antar kalimat. Nilai kesamaan ini dihitung menggunakan Cosine Similarity dengan rumus pada Persamaan 7. Sebelum nilai similaritas dapat dihitung, dilakukan penghitungan perkalian bobot antar kalimat. Ketika nilai i=1, maka bobot vektor dari S_1 akan dikalikan dengan masing-masing bobot dari sembilan kalimat lainnya (S_2, S_3, S_4, S_5, S_6, S_7, S_8, S_9, dan S_{10}).

$$Sim(S_i, S_j) = \frac{\sum(S_i * S_j)}{\sqrt{\sum {S_i}^2}\sqrt{\sum {S_j}^2}} \qquad (7)$$

Berdasarkan seluruh kombinasi perhitungan antar kalimat tersebut, diperoleh hasil Cosine Similarity untuk setiap kalimat yang dapat dilihat pada Tabel 6 berikut.

Tabel 6. Cosine Similarity

S_i	Cosine Similarity
S_1	0.5068
S_2	0.53
S_3	0.5957
S_4	0.6434
S_5	0.4366
S_6	0.2798
S_7	0.5627
S_8	0.3209
S_9	0.3024
S_{10}	0.5318
Rata-Rata	0.471

Menggunakan nilai Cosine Similarity tersebut, dapat ditentukan nilai relevansi awal untuk setiap kalimat. Kalimat dengan bobot di atas rata-rata akan dianggap relevan, sementara sebaliknya dianggap tidak relevan. Dengan rata-rata sebesar 0.471, maka diperoleh 6 kalimat yang relevan (Tabel 7).

Tabel 7. Relevansi Kalimat

S_i	Cosine Similarity	Relevansi
S_4	0.6434	RELEVAN
S_3	0.5957	RELEVAN
S_7	0.5627	RELEVAN
S_{10}	0.5318	RELEVAN
S_2	0.53	RELEVAN
S_1	0.5068	RELEVAN
S_5	0.4366	TIDAK RELEVAN
S_8	0.3209	TIDAK RELEVAN
S_9	0.3024	TIDAK RELEVAN
S_6	0.2798	TIDAK RELEVAN

Relevansi dari suatu kalimat, dapat dilihat dari seberapa relevan tetangga yang ada di sekitarnya. Dengan mempertimbangkan 3 tetangga terdekat (k=3), diperoleh hasil relevansi seperti yang dapat dilihat pada Tabel 8 berikut.

Tabel 8. Relevansi Tetangga

Si	Tetangga | Relevansi	Relevansi
S_4	S_3 | RELEVAN S_7 | RELEVAN S_{10} | RELEVAN	RELEVAN
S_3	S_7 | RELEVAN S_{10} | RELEVAN S_2 | RELEVAN	RELEVAN
S_7	S_{10} | RELEVAN S_2 | RELEVAN S_1 | RELEVAN	RELEVAN
S_{10}	S_2 | RELEVAN S_1 | RELEVAN S_5 | TIDAK RELEVAN	RELEVAN
S_2	S_1 | RELEVAN S_5 | TIDAK RELEVAN S_8 | TIDAK RELEVAN	TIDAK RELEVAN
S_1	S_5 | TIDAK RELEVAN S_8 | TIDAK RELEVAN S_9 | TIDAK RELEVAN	TIDAK RELEVAN
S_5	S_8 | TIDAK RELEVAN S_9 | TIDAK RELEVAN S_6 | TIDAK RELEVAN	TIDAK RELEVAN
S_8	S_9 | TIDAK RELEVAN S_6 | TIDAK RELEVAN S_4 | RELEVAN	TIDAK RELEVAN
S_9	S_6 | TIDAK RELEVAN S_4 | RELEVAN S_3 | RELEVAN	RELEVAN
S_6	S_4 | RELEVAN S_3 | RELEVAN S_7 | RELEVAN	RELEVAN

2.3.6 Hasil Pengujian

Dengan menggunakan 100 buah data uji berupa latar belakang pada laporan Skripsi, diperoleh hasil persentase untuk setiap dokumen sebagai berikut.

Tabel 9. Hasil Pengujian

Dokumen	KNN	K-Means
D1	53.85%	46.15%
D2	58.82%	52.94%
D3	60.87%	52.17%
D4	70.59%	52.94%
D5	63.64%	45.45%
D6	61.90%	52.38%
D7	38.46%	53.85%
D8	62.50%	50.00%
D9	57.89%	52.63%
D10	57.14%	50.00%
D11	50.00%	50.00%
D12	53.33%	50.00%
D13	58.33%	50.00%
D14	57.69%	50.00%
D15	57.69%	50.00%
D16	60.00%	50.00%
D17	61.54%	53.85%
D18	50.00%	50.00%
D19	47.37%	52.63%
D20	40.00%	52.00%
D21	33.33%	53.33%
D22	60.00%	50.00%
D23	47.06%	52.94%
D24	66.67%	50.00%
D25	42.86%	57.14%
D26	15.38%	46.15%
D27	20.00%	60.00%
D28	75.00%	50.00%
D29	26.67%	53.33%
D30	63.64%	45.45%
D31	50.00%	57.14%
D32	64.71%	52.94%
D33	25.00%	50.00%
D34	50.00%	42.86%
D35	60.00%	53.33%
D36	41.67%	50.00%
D37	42.86%	53.57%
D38	52.63%	57.89%
D39	42.86%	50.00%
D40	57.14%	50.00%
D41	43.48%	52.17%
D42	39.13%	52.17%
D43	50.00%	50.00%
D44	16.67%	50.00%
D45	43.75%	50.00%
D46	38.10%	47.62%
D47	36.36%	54.55%
D48	53.33%	53.33%
D49	30.00%	50.00%
D50	38.89%	50.00%
D51	62.50%	50.00%
D52	41.18%	52.94%
D53	36.36%	45.45%
D54	66.67%	50.00%
D55	36.36%	54.55%
D56	38.46%	53.85%
D57	42.86%	47.62%
D58	40.00%	53.33%
D59	61.54%	53.85%
D60	53.85%	53.85%
D61	78.57%	50.00%
D62	70.00%	50.00%
D63	36.36%	54.55%
D64	43.75%	56.25%
D65	42.31%	50.00%
D66	72.73%	54.55%
D67	62.50%	50.00%
D68	45.45%	54.55%
D69	54.55%	54.55%
D70	44.44%	55.56%
D71	70.00%	50.00%
D72	52.63%	52.63%
D73	72.73%	50.00%
D74	45.00%	50.00%
D75	42.11%	52.63%
D76	50.00%	50.00%
D77	54.55%	50.00%
D78	58.33%	50.00%
D79	77.78%	44.44%
D80	66.67%	46.67%
D81	36.36%	45.45%
D82	42.86%	50.00%
D83	57.14%	50.00%
D84	38.10%	52.38%
D85	35.29%	52.94%
D86	34.48%	51.72%
D87	50.00%	37.50%
D88	22.22%	61.11%
D89	50.00%	50.00%
D80	50.00%	60.00%
D91	31.58%	47.37%
D92	50.00%	50.00%
D93	45.45%	54.55%
D94	45.83%	50.00%
D95	21.43%	50.00%
D96	53.85%	46.15%
D97	60.00%	50.00%
D98	50.00%	50.00%
D99	38.89%	50.00%
D100	50.00%	50.00%
Rata-Rata	49.33%	51.14%

3. PENUTUP

3.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, diperoleh hasil akurasi peringkasan sebesar 49% untuk KNN dan 51% untuk K-Means. Walaupun secara umum, KNN memiliki nilai akurasi yang lebih kecil dari K-Means, KNN unggul secara signifikan dibandingkan K-Means di beberapa dokumen. Akurasi K-Means yang dihasilkan juga tergantung dari ukuran dan pusat cluster awal yang terpilih. Maka, sebagai tahap pengembangan selanjutnya, perlu dilakukan penelitian yang membandingkan hasil dari kombinasi nilai ukuran dan pusat cluster awal untuk K-Means dan membandingkan hasil kombinasi K-NN dengan beberapa metode klasifikasi lainnya dalam menentukan parameter kedekatan antar tetangga.

3.1 Saran

Dalam penggunaan dan pengembangan K-NN dan K-Means, pemilihan praproses yang tepat dapat memungkinkan peningkatan akurasi yang signifikan. Maka, berbagai kombinasi praproses seperti Stemming, Stopword Removal, dan POS Tagger dapat dijadikan pertimbangan.

UCAPAN TERIMA KASIH

Penelitian ini didanai oleh Universitas Komputer Indonesia dalam Surat Perjanjian Kerjasama dengan nomor 064BA/SP/LPPM/UNIKOM/V/2017.

DAFTAR PUSTAKA

[1] P. Sudarman, *Menulis di Media Massa*. Yogyakarta: Pustaka Pelajar, 2008.

[2] F. Pratama, "Rancang Bangun Aplikasi Peringkas Teks Otomatis Artikel Berbahasa Indonesia Menggunakan Metode Term Frequency Inverse Document Frequency (TF-IDF) dan K-Means Clustering," Universitas Islam Negeri Sultan Syarif Kasim Riau, 2014.

[3] R. M. Alguliev and R. M. Aliguliyev, "Effective summarization method of text documents," *Proc. - 2005 IEEE/WIC/ACM Int. Web Intell. WI 2005*, vol. 2005, pp. 264–271, 2005.

[4] A. R. Pal, P. K. Maiti, and D. Saha, "An approach to automatic text summarization using simplified lesk algorithm and wordnet," *Int. J. Control Theory Comput. Model.*, vol. 3, no. 4, pp. 15–23, 2013.

[5] R. Al-Hashemi, "Text Summarization Extraction System (TSES) Using Extracted Keywords," *Int. Arab J. e-Technology*, vol. 1, no. 4, 2010.

[6] Y. J. Kumar, O. S. Goh, H. Basiron, N. H. Choon, and P. C. Suppiah, "A review on automatic text summarization approaches," *J. Comput. Sci.*, vol. 12, no. 4, pp. 178–190, 2016.

[7] A. Ridok and T. C. Romadhona, "Peringkas dokumen otomatis menggunakan metode fuzzy model sistem inferensi mamdani," pp. 19–24, 2013.

[8] W. B. Croft, D. Metzler, and T. Strohman, *Search Engines : Information Retrieval in Practice*. USA: Addison-Wesley Publishing Company, 2009.

[9] M. Mustaqhfiri and Z. Abidin, "Peringkasan Teks Otomatis Berita Berbahasa Indonesia Menggunakan Metode Maximum Marginal Relevance," 2001.

[10] C. D. Manning, P. Raghavan, and H. Schutze, "An Introduction to Information Retrieval," 2009.

[11] A. C. Similarity, "Semantic Cosine Similarity," vol. 2, no. 4, pp. 4–5, 2012.

[12] Purnawansyah and Haviluddin, "K-Means clustering implementation in network traffic activities," in *Proceedings - CYBERNETICSCOM 2016: International Conference on Computational Intelligence and Cybernetics*, 2017, pp. 51–54.

[13] O. J. Oyelade, O. O. Oladipupo, and I. C. Obagbuwa, "Application of k Means Clustering algorithm for prediction of Students Academic Performance," *Int. J. Comput. Sci. Inf. Secur.*, vol. 7, no. 1, pp. 292–295, 2010.

[14] T. Jo, "Table based KNN for Text Summarization," pp. 37–42, 2002.

[15] T. M. Mitchell, *Machine Learning*. McGraw-Hill Science/Engineering/Math, 1997.